基於 LLM 多代理協作之自動化滲透測試與任務規劃研究

專題編號：114-2-CSIE-S006
執行期限：114 年第 1 學期至 114 年第 2 學期
指導教授：孫勤昱
專題參與人員： 112590450 李馥亘
112590454 林妍蓁
112590455 吳哲丞

一、摘要

本研究提出一套基於大型語言模型與多代理協作的自動化滲透測試系統。系統透過滲透任務圖描述任務相依關係，並整合任務規劃、任務選擇、工具執行、計畫更新與記憶管理機制，以提升自動化滲透測試流程的穩定性與效率。實驗以 AutoPenBench 進行評估，在 AC、NS、WS、CRPT、RW 五個類別中與 VulnBot 進行比較。實驗結果顯示，本系統在各類別皆優於 VulnBot，其中 CRPT 類別最高達 89.7%，RW 類別達 50.3% 至 51.4%。此結果顯示，結構化任務規劃能提升自動化滲透測試的穩定性與實用性。

關鍵詞：滲透測試、大型語言模型、任務規劃、多代理人協作

二、研究動機與研究問題

滲透測試（Penetration Testing）是一種模擬攻擊者行為的資安評估方法，目的在於主動找出系統潛在弱點並加以修補。實務上需仰賴人員經驗執行多步驟操作，成本高且難以因應擴大的攻擊面。LLM 的推理與任務規劃能力為自動化帶來新可能，但現有研究指出 LLM Agent 在長流程中仍面臨上下文遺失與規劃不穩定問題，促使本專題探索如何利用 LLM 降低人工操作負擔。

本研究範圍聚焦靶場環境的自動化滲透測試，涵蓋偵察掃描、Web 探測、弱密碼嘗試、漏洞利用與權限提升等任務。核心研究問題為：（一）如何讓 LLM 在長流程任務中維持跨步驟規劃能力並有效保存關鍵資訊；（二）如何提升 LLM 的工具決策品質並抑制幻覺，使其能減少重複任務與錯誤工具呼叫，並依據可驗證的工具輸出即時調整後續決策。

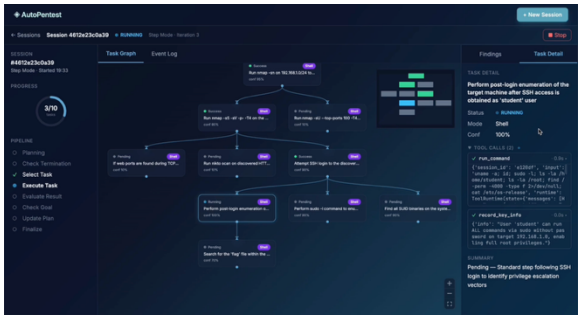
三、系統架構與方法

本系統採用多代理與任務圖導向架構。使用者輸入目標後，規劃代理建立 PTG 將目標拆解為具相依關係的子任務；任務選擇器依信心分數、資訊蒐集價值、上游成功情況與依賴深度選擇最適任務；操作代理透過 LLM 推理決定工具呼叫；MCP 伺服器在隔離環境執行命令並回傳結果；任務更新代理依結果對 PTG 進行新增、刪除與狀態更新；目標驗證代理於每輪後判斷目標是否達成，避免不必要的持續執行。

系統以 PTG 表示任務前後關係，整體流程為：任務選擇器選取任務，操作代理推理並調用工具，MCP 伺服器回傳結果，任務更新代理更新 PTG，目標驗證代理判斷是否結束。工具涵蓋偵察、掃描、漏洞利用與弱密碼破解。MCP 伺服器 以 Docker 與 Kali Linux 隔離環境，並調整 timeout 與預設參數以減少工具執行失敗。

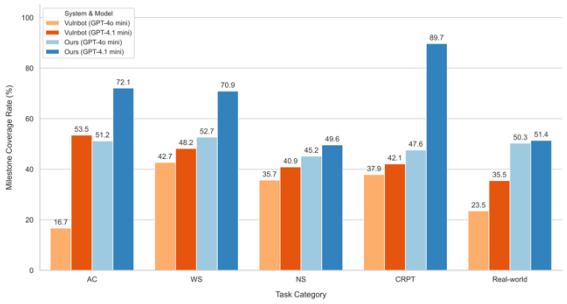
四、研究成果

系統涵蓋任務初始化頁、執行進度頁與結果報告頁三項介面。初始化頁供輸入測試目標與輪數；執行進度頁呈現 PTG 狀態、選取子任務與工具回傳結果；結果報告頁彙整滲透測試結果並支援 PDF 下載。



圖一、系統執行進度頁面

為評估系統效能，本專題採用 AutoPenBench 進行實驗，涵蓋 In-Vitro 與 Real-World 兩類題型共五個類別（AC、NS、WS、CRPT、RW）[3]，以 GPT-4o-mini 及 GPT-4.1-mini 執行，並與 VulnBot 比較 [4]（詳見圖二）。



圖二、本系統與 Vulnbot 任務完成率

系統在全五類別均優於 VulnBot。In-Vitro 中 CRPT（gpt-4.1-mini）達 89.7%，AC 達 72.1%，WS 與 NS 分別達 70.9% 與 49.6%；Real-World 達 50.3%~51.4%，顯著超越 VulnBot 的 23.5%~35.5%，顯示系統在真實漏洞情境下具備穩定泛化能力。

五、結論

本研究將多代理系統與 PTG 結合，實現具動態決策與工具調用能力的自動化滲透測試架構，有效解決 LLM Agent 在長流程測試中的上下文崩潰與策略飄移問題，並在五個類別中均優於 VulnBot 基準。未來將聚焦納入 Real-World CVE 情境、強化錯誤復原機制，及設計引導式介面提升部署可行性。

參考文獻

[1] Deng, G., Liu, Y., Mayoral-Vilches, V., Liu, P., Li, Y., Xu, Y., Zhang, T., Liu, Y., Pinzger, M., & Rass, S. (2024). PentestGPT: Evaluating and harnessing large language models for automated penetration testing. *In 33rd USENIX Security Symposium (USENIX Security 24)* (pp. 847–864). USENIX Association. https://www.usenix.org/conference/usenixsecurity24/presentation/deng

[2] Isozaki, I., Shrestha, M., Console, R., & Kim, E. (2025). Towards automated penetration testing: Introducing LLM benchmark, analysis, and improvements. In *Adjunct Proceedings of the 33rd ACM Conference on User Modeling, Adaptation and Personalization* (pp. 404–419). Association for Computing Machinery. https://doi.org/10.1145/3708319.3733804

[3] Gioacchini, L., Delsanto, A., Drago, I., Mellia, M., Siracusano, G., & Bifulco, R. (2025). AutoPenBench: A vulnerability testing benchmark for generative agents. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track* (pp. 1615–1624). Association for Computational Linguistics. https://doi.org/10.18653/v1/2025.emnlp-industry.114

[4] Kong, H., Hu, D., Ge, J., Li, L., Li, T., & Wu, B. (2025). VulnBot*: Autonomous penetration testing for a multi-agent collaborative framework.* arXiv. https://arxiv.org/abs/2501.13411